Graphite – Relációs adatanalízis gráfokkal a felhőben

Lévai Zalán Bálint
Felkészítő tanár: Bärnkopf Bence
Városmajori Gimnázium,
1122, Budapest, Városmajor u. 71

1. Bevezetés

A jelenleg elérhető adatanalizáló eszközök nem fordítanak kellő figyelmet az adatok közt fennálló kapcsolatok, azok egymásra vonatkozó relációjának megjelenítésére, így a sokszor legértékesebbnek mondható információk kerülnek látókörön kívül. Projektem többek között ezen problémát célozza megoldani; olyan felületet kínál adathalmazokhoz, melyben az adatelemek relációja elsődleges szerepet kap. Ez ideális számos területen előkerülő problémák esetén; legyen szó szociális hálókról, számítógépes hálózatokról, üzleti menedzsment megoldásokról vagy logisztikai rendszerekről.

Továbbá az eddigi megoldások körében megoldatlan problémának bizonyult az egyidejű munkavégzés. Míg a klasszikus dokumentumok csoportos szerkesztésére számos lehetőség létezik, addig ez nem megoldott az adatelemzés terén; ezzel jelentősen lassítva a munkavégzést.

2. Probléma megoldásának menete

2.1. A gráfmotor; *Graphite Core*

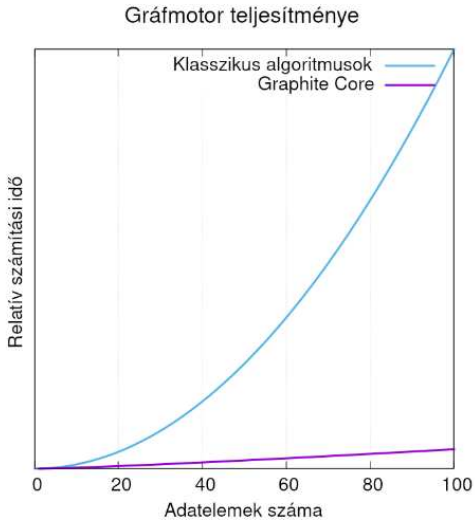
A projekt magját képező, fizikai együtthatásokon alapuló, erő-vezérelt gráfok kezelésére alkalmas gráfmotor a *Graphite Core*. Tervezése során elsődleges szempont volt a nagy teljesítmény elérése, valamint az algoritmusok maximális párhuzamosítottsága, és szálasíthatósága. Ennek érdekében a *Graphite Core* egyebek mellett egy saját, parallelizált, Barnes-Hut szimulációs algoritmus implementációt alkalmaz, amely az átlagos n-test fizikai szimulációkkal szemben messzemenőkig gyorsabbnak és kisebb számítási komplexitásúnak bizonyul, amely az *1. ábrán* is látható.

Míg a klasszikus fizikai motorok algoritmusai *(1)* komplexitásúak, addig a *Graphite Core* által használté ezeknél jelentősen kevesebb, *(2)* komplexitású.

$$O(n^2) \tag{1}$$

$$O(n \times \log n) \tag{2}$$

Ennek eredménye, hogy a gráfok valós idejű kezelése több ezer adatelem esetén is gyors és használható marad.



1. ábra: A *Graphite Core* relatív teljesítménye az átlagos számítási idő szerint

A teljesítmény maximalizálása mellett kiemelt szerepet kapott a gráfmotor fejlesztése során annak bővíthetősége, modularizálása. Ezt példázza többek között annak saját interfész nyelve, amely a megszokott szűrő query-ken kívül számos egyéb feladat ellátására képes; hozzáférést nyújt a felhasználók számára a *Graphite Core* teljes API felületéhez.

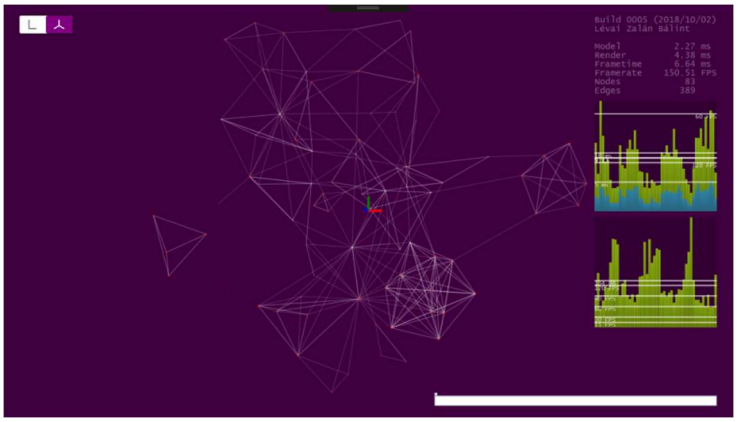
2.2. Az első prototípus

A projekt egy Windows-on futtatható kliens alkalmazásként öltött először formát, melynek fejlesztése 2018 októberében kezdődött. Ekkor ment végbe a gráfmotor fejlesztésének jelentős része is.

Ez, a .NET keretrendszer és a Windows Presentation Foundation (WPF) segítségével megírt prototípus számos funkcionalitásnak adott teret. Megjelennek benne a máig is használt kettő- és többdimenziós gráf nézetek (2. ábra), amelyek a böngészést és a szűrést segítik elő.

Továbbá lehetőséget nyújt a Graphite Core által nyújtott interfész nyelvhez tartozó parancsértelmező használatára. Mindezt kiegészíti a programfejlesztő környezetekből ismertekhez hasonló intelligens, kontextuális parancs javaslattokkal (3. ábra).

Bár később a kliens oldalon futtatott számításokra vonatkozó terveket felváltotta egy felhő alapú számítási modell - többek között a kliens gépek leterheltségének csökkentése érdekében -, az itt feltüntetett funkcionalitás meghatározó volt a további fejlesztésekre nézve is.



2. ábra: 3D-s gráfnézet

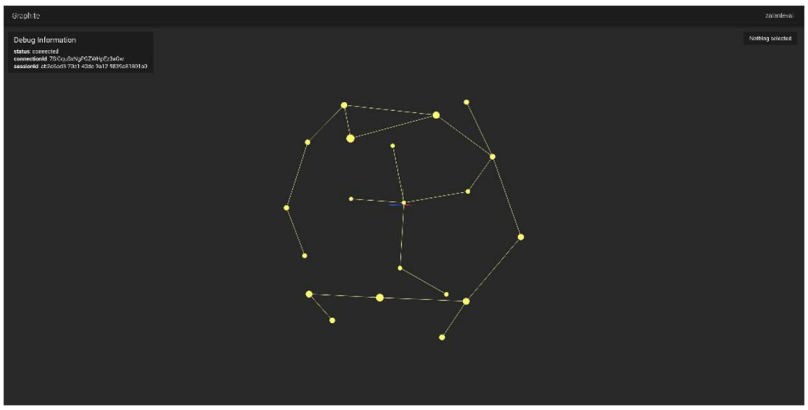
3. ábra: Parancsértelmező kontextuális javaslatokkal

2.3. Felhő és Web

A projekt fejlesztésének következő és jelenlegi állomását a szerver oldali számításokon alapuló webes alkalmazás képezi. Erre a változtatásra 2019 januárjában került sor. Ennek a változtatásnak részben a platform függetlenség hiánya, a helyi alkalmazások telepítésével járó komplikációk és költségek elkerülése, valamint a korábban említett együttműködési lehetőségek felmerülése adott okot. A felhő alapú modell ezekre mind megoldást nyújt és lehetőséget ad az együttműködési lehetőségek megvalósítására.

A szerver oldali megvalósításhoz az ASP.NET Core keretrendszer használatával került sor. Az architekturális váltás megvalósításának legnehezebb lépése a szerver és kliens kommunikációja volt; erre megoldást nyújtott a WebSocket alapú SignalR protokoll, amely lehetővé teszi az alkalmazás által elvárt magas frissítési rátájú és közel valós idejű kommunikációt a felek között.

Ezzel lehetővé vált a korábban említett egyidejű munkamenetek megvalósítása.



4. ábra: Web szerkesztő felülete

A kliens oldalon egy modern alapokra épülő webalkalmazás működik. Megvalósítására a React komponens könyvtár és a TypeScript nyelv segítségével kerül sor; a szerkesztő pedig a WebGL-en alapszik.

3. Elért eredmények

A webes platform jelenlegi állapotában közel funkció kompatibilis az eredeti kliens alkalmazással, valamint bizonyos területeken messzemenőkig meghaladja azt. Képes teljesen valós időben a munkamenetek kezelésére, a gráfokon végzett módosítások és egyéb műveletek szinkronban tartására.

Természetesen a projekt korai fázisa, valamint egyedüli fejlesztése miatt az analízis eszközök még számottevő része nem került megvalósításra, valamint a felhasználói felület is sok területen hiányokban szenved. Mindennek ellenére ezek jelentős részének technológiai háttere már biztosított a kódbázis jelenlegi formájában is.

A Graphite sok szempontból egyedülálló szoftver a kategóriájában; a korábban nehezen vizsgálható információk és összeköttetések kinyerésére ad lehetőséget, amely számos területen hasznos eszközzé teszi. A kódbázis is ezt reflektálja; könnyedén bővíthető és modularizált.